

## Contents

<b>1 Routine/Function Prologues</b>	<b>2</b>
1.0.1 getgeos.F90 (Source File: getgeos.F90) . . . . .	2
1.1 Core Functions of getgeos . . . . .	2
1.1.1 geosfile (Source File: getgeos.F90) . . . . .	6

## 1 Routine/Function Prologues

### 1.0.1 getgeos.F90 (Source File: getgeos.F90)

Opens, reads, and interpolates GEOS forcing.

TIME1 = most recent past data

TIME2 = nearest future data

The strategy for missing data is to go backwards up to 10 days to get forcing at the same time of day.

### 1.1 Core Functions of getgeos

**tick** Determines GEOS data times

**geosfile** Puts together appropriate file name for 3 hour intervals

**readgeos** Interpolates GEOS data to LDAS grid

### REVISION HISTORY:

```

1 Oct 1999: Jared Entin; Initial code
25 Oct 1999: Jared Entin; Significant F90 Revision
11 Apr 2000: Brian Cosgrove; Fixed name construction error
              in Subroutine ETA6HRFIL
27 Apr 2000: Brian Cosgrove; Added correction for use of old shortwave
              data with opposite sign convention from recent shortwave data.
              Added capability to use time averaged shortwave & longwave data
              Altered times which are passed into ZTERP--used to be GMT1
              and GMT2, now they are LDAS%ETATIME1 and LDAS%ETATIME2
30 Nov 2000: Jon Radakovich; Initial code based on geteta.f
17 Apr 2001: Jon Gottschalck; A few changes to allow model init.
13 Aug 2001: Urszula Jambor; Introduced missing data replacement.
5 Nov 2001: Urszula Jambor; Reset tiny negative SW values to zero.

```

### INTERFACE:

```
subroutine getgeos()
```

### USES:

```

use lisdrv_module, only : lis
use time_manager
use spmdMod
use tile_spmdMod
use time_module, only : tick
use baseforcing_module, only: glbdata1,glbdata2
use geosdomain_module, only : geosdrv
use def_ipMod, only : def_ip_input

```

### CONTENTS:

```

if ( masterproc ) then
    nstep = get_nstep()
endif
#if ( defined OPENDAP )
    call MPI_BCAST(nstep,1,MPI_INTEGER,0,MPI_COMM_WORLD,ierr)
#endif
!-----
! Determine the correct number of forcing variables
!-----
if ( nstep .eq. 0) then
    nforce = geosdrv%nmif
else
    nforce = lis%f%nf
endif
lis%f%findtime1=0
lis%f%findtime2=0
movetime=0
!-----
! Determine Required GEOS Data Times
! (The previous hour & the future hour)
!-----
yr1=lis%t%yr      !Time now
mo1=lis%t%mo
da1=lis%t%da
hr1=lis%t%hr
mn1=lis%t%mn
ss1=0
ts1=0

call tick(timenow,doy1,gmt1,yr1,mo1,da1,hr1,mn1,ss1,ts1)

yr1=lis%t%yr      !Previous Hour
mo1=lis%t%mo
da1=lis%t%da
hr1=3*((lis%t%hr)/3)
mn1=0
ss1=0
ts1=0
call tick(time1,doy1,gmt1,yr1,mo1,da1,hr1,mn1,ss1,ts1)

yr2=lis%t%yr      !Next Hour
mo2=lis%t%mo
da2=lis%t%da
hr2=3*((lis%t%hr)/3)
mn2=0
ss2=0
ts2=3*60*60

```

```

call tick(time2,doy2,gmt2,yr2,mo2,da2,hr2,mn2,ss2,ts2)
if(timenow.gt.geosdrv%geostime2) then
  movetime=1
  lis%f%findtime2=1
endif

if ( nstep.eq.0 .or. nstep.eq.1 .or.lis%f%rstflag.eq.1 ) then
  lis%f%findtime1=1
  lis%f%findtime2=1
  glbdata1 = 0
  glbdata2 = 0
  movetime=0
  lis%f%rstflag = 0
endif
lis%f%shortflag=2           !Time averaged SW
lis%f%longflag=2            !Time averaged LW

if(time1>geosdrv%griduptime.and.geosdrv%gridchange) then
  print*, 'Time change..., Switching to GEOS4'
  geosdrv%ncold = 288
!-----
! Reinitialize the weights and neighbors
!-----
kgdsi = 0
kgdsi(1) = 0
kgdsi(2) = geosdrv%ncold
kgdsi(3) = geosdrv%nrold
kgdsi(4) = -90000
kgdsi(7) = 90000
kgdsi(5) = -180000
kgdsi(6) = 128
kgdsi(8) = 179000
kgdsi(9) = 1000
kgdsi(10) = 1000
kgdsi(20) = 255
call def_ip_input(kgdsi)
geosdrv%gridchange = .false.

if ( lis%f%ecor == 1 ) then
  lis%f%gridchange = 1
  elevfile = lis%p%elevfile
  c = index(elevfile,"geos3")
  fpart1 = elevfile(1:c+3)
  fpart2 = elevfile(c+5:40)
  lis%p%elevfile = trim(fpart1) // "4" // trim(fpart2)
  print*, 'Use newer elevation difference file: ', lis%p%elevfile
  print*, 'Transitioned from GEOS3 to GEOS4 grid dimensions.'
  call get_geos4_diff()

```

```
        endif
    endif
!-----
! Establish geostime1
!-----
if (lis%f%findtime1==1) then
    order=1
    ferror = 0
    try = 0
    ts1 = -24*60*60
    do
        if ( ferror /= 0 ) then
            exit
        end if
        try = try+1
        call geosfile(name,geosdrv%geosdir,yr1,mo1,da1,hr1,geosdrv%ncold)
        call readgeos(order,name,lis%d,lis%t%tscount,ferror)
        if ( ferror == 1 ) then
!-----
! successfully retrieved forcing data
!-----
            geosdrv%geostime1=time1
        else
!-----
! ferror still=0, so roll back one day
!-----
            call tick(dumbtime1,doy1,gmt1,yr1,mo1,da1,hr1,mn1,ss1,ts1)
        end if
        if ( try > ndays ) then
            print *, 'ERROR: GEOS data gap exceeds 10 days on file 1'
            call endrun
        end if
        end do
    endif
    if(movetime.eq.1) then
        geosdrv%geostime1=geosdrv%geostime2
        lis%f%findtime2=1
        do f=1,nforce
            do c=1,lis%d%ngrid
                glbdata1(f,c)=glbdata2(f,c)
            enddo
        enddo
    endif
    if(lis%f%findtime2.eq.1) then
        order=2
        ferror = 0
        try = 0
        ts2 = -24*60*60
```

```

do
  if ( ferror /= 0 ) exit
  try = try+1
  call geosfile(name,geosdrv%geosdir,yr2,mo2,da2,hr2,geosdrv%ncold)
  call readgeos(order,name,lis%d,lis%t%tscount,ferror)

  if ( ferror == 1 ) then
!---
! successfully retrieved forcing data
!---
      geosdrv%geostime2=time2
    else
!---
! ferror still=0, so roll back one day
!---
      call tick(dumbtime2,doy2,gmt2,yr2,mo2,da2,hr2,mn2,ss2,ts2)
    end if
    if ( try > ndays ) then
      print *, 'ERROR: GEOS data gap exceeds 10 days on file 2'
      call endrun
    end if
  end do
endif

84 format('now',i4,4i3,2x,'pvt ',a22,' nxt ',a22)
! if ((lis%f%gridchange==1).and.(geosdrv%ncold==288)) then
!   lis%f%gridchange=0
! endif
  return

```

---

### 1.1.1 geosfile (Source File: getgeos.F90)

This subroutine puts together GEOS file name

INTERFACE:

```

subroutine geosfile(name,geosdir,yr,mo,da,hr, ncold)

implicit none

```

*INPUT PARAMETERS:*

```

character*40 geosdir
integer yr,mo,da,hr,ncold

```

*OUTPUT PARAMETERS:*

```
character*80 name
```

## CONTENTS:

```
91 format(a4,i3,a11,i3)
92 format(80a1)
93 format(a80)
94 format(i4,i2,i2,a2)
95 format(10a1)
96 format(a40)
97 format(a8)
98 format(a1,i4,i2,a1)
99 format(8a1)
```

---

```
!-----  
! Make variables for the time used to create the file  
! We don't want these variables being passed out
```

---

```
!-----  
uyr=yr  
umo=mo  
uda=da  
uhr = 3*(hr/3) !hour needs to be a multiple of 3 hours
```

---

```
!-----  
! Determine initcode for the hour of the forecast file  
! If the time is 12 or later the file is time stamped  
! with the next day. So check for that first
```

---

```
if(uhr<3)then
    initcode = '00'
elseif(uhr<6)then
    initcode = '03'
elseif(uhr<9)then
    initcode = '06'
elseif(uhr<12)then
    initcode = '09'
elseif(uhr<15)then
    initcode = '12'
elseif(uhr<18)then
    initcode = '15'
elseif(uhr<21)then
    initcode = '18'
elseif(uhr<24)then
    initcode = '21'
endif
```

```
write(UNIT=temp,FMT='(A40)') geosdir
read(UNIT=temp,FMT='(80A1)') (fbase(i),i=1,80)
```

```
write(UNIT=temp,FMT='(a1,i4,i2,a1)') '/', uyr, umo, '/'
```

```
read(UNIT=temp,FMT='(8A1)') fdir
do i=1,8
    if(fdir(i).eq.( ' )) fdir(i)='0'
enddo

write(UNIT=temp,FMT='(i4,i2,i2,a2)') uyr,umo,uda,initcode
read(UNIT=temp,FMT='(10A1)') ftime
do i=1,10
    if(ftime(i).eq.( ' )) ftime(i)='0'
enddo

if(ncold==360) then
    write(UNIT=temp,FMT='(A8)') '.GEOS323'
    read(UNIT=temp,FMT='(80A1)') (fsubs(i),i=1,8)
else
    write(UNIT=temp,FMT='(A6)') '.GEOS4'
    read(UNIT=temp,FMT='(80A1)') (fsubs(i),i=1,6)
endif
c=0
do i=1,80
    if(fbase(i).eq.( ' ).and.c.eq.0) c=i-1
enddo

if (ncold==360) then
    write(UNIT=temp,FMT='(80a1)') (fbase(i),i=1,c),(fdir(i),i=1,8), &
        (ftime(i),i=1,10),(fsubs(i),i=1,8)
else
    write(UNIT=temp,FMT='(80a1)') (fbase(i),i=1,c),(fdir(i),i=1,8), &
        (ftime(i),i=1,10),(fsubs(i),i=1,6)
endif

read(UNIT=temp, FMT='(a80)') name
write(*,*)hr,name
return
```